REMARKS

The following is intended as a full and complete response to the Final Office Action dated December 23, 2008. Claims 1, 2, 4-8, and 10-22 and 24-31 were examined. Applicants traverse the Examiner's rejections of the pending claims for the reasons set forth below.

35 U.S.C. § 102(e) Rejections

Claims 1, 2, 4-8, 10-14 are rejected under 35 U.S.C. § 102(a) as being anticipated by Elzur (US 7,346,701).

Claim 8

Claim 8 recites the limitations of determining that the first frame and the second frame are out-of-sequence based on comparing the sequence number stored in the entry with a sequence number in the second frame, and storing a flag in the entry to indicate that synchronization is requested for the connection. Elzur fails to teach comparing the sequence number stored in the entry with a sequence number in the second frame, and storing a flag in the entry to indicate that synchronization is requested for the connection. Nowhere does Elzur teach that synchronization is requested for a connection.

The Examiner relies on Figure 4 of Elzur for teaching the limitation of storing a flag in the entry to indicate that synchronization is requested for the connection. Figure 4 illustrates an IP datagram header format that includes a field labeled "flag." The IP datagram is received by the NIC described by Elzur. Elzur is completely silent regarding the flag field. Nowhere does Elzur teach or suggest that the flag is stored in the delegated connection table. Furthermore, there is no teaching or suggestion whatsoever that the flag is associated with synchronization. The presence of the word "flag" in Figure 4 does not rise to the level of a teaching as required by 35 U.S.C. § 102(e).

Since Elzur fails to teach each and every limitation recited in claim 8, this claim cannot be anticipated by Elzur.

Claim 10

Claim 10 recites the limitations of uploading the payload data of the first frame to at least one legacy buffer that is in a first portion of the memory allocated to a driver configured to interface between the offload unit and an application program when a user buffer in a

second portion of the memory that is allocated to the application program is not available. Elzur does not teach or suggest these limitations.

Elzur teaches "copying data of an incoming packet to a host resident buffer..." (see column 15, lines 1-7). The host resident buffer is located in the host memory and is not equivalent to an entry of a delegated connection table that is stored in the offload unit. Elzur is silent regarding the condition of a user buffer not being available. Elzur expressly teaches that the host buffers may be pre-posted application buffers (see column 15, lines 30-34). Those skilled in the art recognize that the portion of system memory that is allocated to an application program should not also be allocated to a driver. Nowhere does Elzur teach or suggest that data is uploaded to buffers allocated to a driver when the host buffers are no available.

Since Elzur fails to teach each and every limitation recited in claim 10, this claim cannot be anticipated by Elzur.

Claims 1, 2, 4-8, 10-14

Claim 1 recites the limitation of selecting, by a TCP stack, a connection for processing by an offload unit. Elzur fails to teach or suggest these limitations.

The TCP offload apparatus of Elzur processes all TCP packets, buffers portions of the packets in elastic buffers, and places portions of the packets in host memory. Nowhere does Elzur teach or suggest that a connection is selected by a TCP stack for offload processing, as claimed. In contrast, each and every incoming TCP frame is associated with a connection, and when a frame is received by the offload apparatus, the connection that the frame is associated with is identified (see step 120 of Figure 11). Elzur teaches that a cache may be used to store the most active connections since storing all of the connection information within the offload apparatus may be costly (see col. 12, line 65 - col. 13 line 1). Those skilled in the art understand that the presence or absence of data is determined by access frequency rather than by having a TCP stack select connections for storage in the cache.

Since Elzur fails to teach or suggest the limitation that any connection is selected by the TCP stack for processing by the offload unit, claim 1 and dependent claims 2, 4-8, and 10-14 cannot be anticipated by Elzur.

Furthermore, claim 1 recites the limitations of storing a sequence number representing a next expected sequence number in an entry of the delegated connection table when a first

frame is received and reading the entry when a second frame is transmitted. Elzur fails to teach or suggest these limitations as well.

Elzur teaches that TCP connection context is fetched for a received frame (see step 130 of Figure 11). Elzur teaches that context information is stored in a memory and accessed to process frames for a connection (see col. 10, lines 46-48 and lines 59-65 and Figures 12, 13, and 14). However, Elzur does not describe the specific connection state that is included in the context information. Elzur describes constructing a mapping between TCP sequence numbers and the host buffers (see column 15, lines 14-20) when frames are received. As shown in Figure 9, the mapping occurs through a buffer descriptor table and is used as an offset to a base sequence number to determine a physical address of a host buffer. Nowhere does Elzur teach or suggest storing a next expected sequence number in an entry of the mapping information. Furthermore, Elzur provides no teaching whatsoever that the same entry of the mapping information is read when a second frame is transmitted.

Since Elzur fails to teach each and every limitation recited in claim 1, this claim and dependent claims 2, 4-8, and 10-14 cannot be anticipated by Elzur.

In addition to the foregoing, claim 12 recites the limitations of uploading any subsequent frames received for the connection to one or more additional legacy buffers until resynchronization is signaled by the TCP stack. Elzur fails to teach any type of resynchronization. The Examiner states that the limitations of claim 12 are described in lines 4-9, without specifying a column number. However, Applicant is unable to find any description of synchronization or resynchronization in Elzur. Therefore, claim 12 is in condition for allowance independent of its dependency on allowable claim 1.

Claims 13 and 14 recite the limitations of invalidating any buffer descriptors for portions of the memory that are available for storing data received on the connection. Clearly the buffer descriptors indicate where frame data that is received by the offload unit is stored in memory. The Examiner relies on extracting validity information from a TCP header that is received by the offload apparatus for teaching this limitation (see col. 12, lines 36-39). The buffer descriptors are not included in the header or derived from the header. Nowhere does Elzur teach invalidating host buffers or any buffer descriptors. Therefore, claims 13 and 14 are in condition for allowance independent of their dependency on allowable claim 1.

35 U.S.C. § 103(a) Rejections

Claims 15-22 and 24-31 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Elzur in view of Odenwald (US 6,310,884).

Claim 18

Claim 18 recites the limitations of setting a request buffer flag in the delegated connection table when a user buffer is not available. Elzur and Odenwald each fail to teach or suggest these limitations. The Examiner relies on Elzur's description of allocated buffers (see col. 15, lines 1-9) for teaching or suggesting this limitation. Again, Elzur teaches only that the buffers may be pre-allocated. Nowhere does Elzur teach or suggest that a flag is set in a delegated connection table when a buffer is not available. Odenwald teaches only a method of transmitting data blocks to improve buffer allocation within the receiver and therefore fails to cure the deficiencies of Elzur. Consequently, the combination of these two references fails to teach or suggest each and every limitation of claim 18.

Claims 15-22, 24-28, and 30-31

Claims 15 and 22 recite the limitation that the delegated connections are selected by a transmission control protocol (TCP) stack for processing by an offload unit that includes the delegated connection table. As previously described with regard to claim 1, Elzur fails to teach or suggest the limitation of a TCP stack selecting the delegated connections. Odenwald teaches only a method of transmitting data blocks to improve buffer allocation within the receiver and, therefore, fails to cure the deficiencies of Elzur relative to claims 15 and 22. For these reasons, the combination of Elzur and Odenwald can render amended claims 15 and 22 or dependent claims 16-21, 24-28, and 30-31 obvious.

Furthermore, claims 15 and 22 recite the limitations of the delegated connection table storing an acknowledgment (ACK) number, timestamp data, and a count of unacknowledged frames in an entry. As previously explained with regard to amended claim 1, Elzur describes constructing a mapping between TCP sequence numbers and the host buffers and cacheing the most active connections within the offload apparatus. Nowhere does Elzur teach or suggest storing an acknowledgment (ACK) number, timestamp data, and a count of unacknowledged frames in an entry of the mapping information or the cache. Timer 220 (see Fig. 10) of Elzur tracks transmit and retransmit events and is not stored in the mapping information or the cache. Elzur simply does not teach or suggest that a delegated connection

table stores an acknowledgment (ACK) number, timestamp data, and a count of unacknowledged frames in an entry, as recited in claims 15 and 22.

The Examiner relies on Odenwald for teaching storing a count of unacknowledged frames in the entry of the delegated connection table. Odenwald teaches receiving a count of transmitted frames, SEQ_CNT. The SEQ_CNT is not functionally equivalent to the count of unacknowledged frames. As known by those skilled in the art, frames that are successfully received are acknowledged. Therefore, the count of unacknowledged frames is the number of frames that are received, but not yet acknowledged. Clearly, the count of unacknowledged frames cannot be equal to the count of transmitted frames that have been received. Further, Odenwald fails teach or suggest storing the SEQ_CNT. In Figure 5 Odenwald shows an acknowledgement number, but Odenwald fails to teach or suggest storing the acknowledgement number. The illustration of a protocol header does not rise to the level of a teaching required to anticipate or render obvious the pending claims. For these reasons, no combination of Elzur and Odenwald can render amended claims 15 and 22 or dependent claims 16-21, 24-28, and 30-31 obvious.

Claim 29

Claim 29 recites the limitations of updating connection state data that includes clearing an unACKnowledged count and updating the ACK number with a last ACKnowledged number. As previously explained with regard to claim 15, Elzur fails to teach or suggest storing any acknowledgment (ACK) number or a count of unacknowledged frames in an entry of the mapping information or the cache. Odenwald fails to cure these deficiencies of Elzur. For these reasons, claim 29 is in condition for allowance independent of its dependency on allowable claim 1.

CONCLUSION

Based on the above remarks, Applicants believe that they have overcome all of the rejections set forth in the Final Office Action mailed on December 23, 2008 and that the pending claims are in condition for allowance. If the Examiner has any questions, please contact the Applicant's undersigned representative at the number provided below.

Respectfully submitted,

Stephanie Winner
Registration No. 52,371
PATTERSON & SHERIDAN, L.L.P.
3040 Post Oak Blvd. Suite 1500
Houston, TX 77056
Telephone: (713) 623-4844
Facsimile: (713) 623-4846
Agent for Applicants